UVM Tips and Tricks
- Runtime Tips

Presented by Srivatsa Vasudevan - Synopsys, Inc.

Slides by Srinivasan Venkataramanan, VerifWorks
UVM TB Hierarchy

- uvm_test_top
- uvm_env
- uvm_agent
  - uvm_driver
  - uvm_sequencer
  - uvm_monitor
  - uvm_scoreboard
Factory Pattern + Constraint Example

Program

Environment

Monitor components → scoreboard

Transactions

s2p_xactn

Command component
Simulation

# UVM_INFO @ 0: reporter [RNTST] Running test factory_test...

# UVM_INFO @ 0: reporter [] Doing build

# UVM_INFO @ 0: uvm_test_top [FACTORY] Overriding s2p_xactn with pattern testcase

# UVM_INFO @ 0: uvm_test_top.env_0.agent0.driver [S2P_DRIVER] S2P_DRIVER : item sent is

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Size</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>s2p_xactn_with_pld</td>
<td>integral</td>
<td>32</td>
<td>'hff</td>
</tr>
<tr>
<td>pkt_pld</td>
<td>integral</td>
<td>32</td>
<td></td>
</tr>
</tbody>
</table>
set_type_override_by_type(
    .original_type
        (s2p_xactn::get_type()),
    .override_type(
        s2p_err_xn::get_type()));

// ... some time later
set_type_override_by_type(
    .original_type
        (s2p_xactn::get_type()),
    .override_type(
        s2p_new_err_xn::get_type())
    .replace(1));

A while later..

1st override

Default - replace
Multiple Overrides - Replace

- Sample log
- ID: TPREGR

[TPREGR] Original object type 's2p_xactn' already registered to produce 's2p_err_xactn'. Replacing with override to produce type 's2p_new_err_xn'.
Multiple Overrides - Ignore

```cpp
set_type_override_by_type(
  .original_type
    (s2p_xactn::get_type()),
  .override_type(
    s2p_err_xn::get_type()));
// ... some time later
set_type_override_by_type(
  .original_type
    (s2p_xactn::get_type()),
  .override_type(
    s2p_new_err_xn::get_type())
  .replace(0));
```

1st override

A while later..

replace = 0
Multiple Overrides - Ignore

• Sample log
• ID: TPREGRD

[TPREGD] Original object type 's2p_xactn' already registered to produce 's2p_err_xactn'. Set 'replace' argument to replace the existing entry.
Replace in Instance Specific Override?

- No “replace” feature in instance specific API
- No warning/information from UVM BCL either!
Factory Debug

• UVM has built-in debug features for factory
• uvm_factory::print()
  – Not a static method though!

```cpp
virtual function void end_of_elaboration_phase (uvm_phase phase);

uvm_factory f;

f = uvm_factory::get();
f.print();
f.print(.all_types(0));
f.print(.all_types(2));
```
Factory Debug - Print

- **all_types = 1 (Default)**

```plaintext
### Factory Configuration (*)

<table>
<thead>
<tr>
<th>Requested Type</th>
<th>Override Path</th>
<th>Override Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>s2p_xactn</td>
<td>uvm_test_top.env_0.agent0.sequencer.*</td>
<td>s2p_xactn_with_pld_pattern</td>
</tr>
<tr>
<td>s2p_xactn</td>
<td>uvm_test_top.env_0.agent0.sequencer.*</td>
<td>s2p_xactn_extra</td>
</tr>
</tbody>
</table>

No type overrides are registered with this factory.

All types registered with the factory: 51 total
(types without type names will not be printed)

<table>
<thead>
<tr>
<th>Type Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>factory_test</td>
</tr>
<tr>
<td>s2p_agent</td>
</tr>
<tr>
<td>s2p_base_test</td>
</tr>
<tr>
<td>s2p_driver</td>
</tr>
<tr>
<td>s2p_env</td>
</tr>
<tr>
<td>s2p_fcov</td>
</tr>
<tr>
<td>s2p_par_mon</td>
</tr>
<tr>
<td>s2p_rand_seq</td>
</tr>
<tr>
<td>s2p_scoreboard</td>
</tr>
<tr>
<td>s2p_sequencer</td>
</tr>
<tr>
<td>s2p_ser_mon</td>
</tr>
<tr>
<td>s2p_xactn</td>
</tr>
<tr>
<td>s2p_xactn_extra</td>
</tr>
</tbody>
</table>
```

Registered user types
## Factory Debug - Print

- all_types = 0 (Recommended by VerifLabs, CVC)

```
### Factory Configuration (*)

Instance Overrides:

<table>
<thead>
<tr>
<th>Requested Type</th>
<th>Override Path</th>
<th>Override Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>s2p_xactn</td>
<td>uvm_test_top.env_0.agent0.sequencer.*</td>
<td>s2p_xactn_with_pld_pattern</td>
</tr>
<tr>
<td>s2p_xactn</td>
<td>uvm_test_top.env_0.agent0.sequencer.*</td>
<td>s2p_xactn_extra</td>
</tr>
</tbody>
</table>

No type overrides are registered with this factory
```

Overrides alone!
Factory Debug - Print

- all_types = 2 (Includes uvm_* too)

```plaintext
### Factory Configuration (*)

Instance Overrides:

<table>
<thead>
<tr>
<th>Requested Type</th>
<th>Override Path</th>
<th>Override Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>s2p_xactn</td>
<td>uvm_test_top.env_0.agent0.sequencer.*</td>
<td>s2p_xactn_with_pld_pattern</td>
</tr>
<tr>
<td>s2p_xactn</td>
<td>uvm_test_top.env_0.agent0.sequencer.*</td>
<td>s2p_xactn_extra</td>
</tr>
</tbody>
</table>

No type overrides are registered with this factory

All types registered with the factory: 51 total (types without type names will not be printed)

```

Overrides +

Registered user types +

UVM_*
Factory Undo? Anyone?

- Allowed in UVM 1.2
- Needs a work-around in UVM 1.1d
A while later...

```cpp
set_type_override_by_type(
    .original_type
      (s2p_xactn::get_type()),
    .override_type(  
      s2p_err_xn::get_type()));

set_type_override_by_type(
    .original_type
      (s2p_xactn::get_type()),
    .override_type(  
      s2p_xactn::get_type()));
```
Undo – Error in UVM 1.1d

• Sample log
• ID: **TYPDUP**

[TYPDUP] Original and override type arguments are identical: s2p_xactn
uvm_component::new()

- Consistent, pre-defined prototype for `new()` for all components
  - `name` (string)
  - `parent` (uvm_component)
- All components – driver, sequencer, monitor, etc. SHOULD use this

```plaintext
class s2p_driver extends uvm_driver #(s2p_xactn);

function new(string name,
              uvm_component parent);
    super.new(name, parent);
endfunction : new
```
s2p_agent::build_phase()

- Use factory pattern – type_id::create()

```cpp
s2p_driver_0 = s2p_driver::type_id::create("s2p_driver_0", this);

s2p_sqr_0 =
    s2p_sequencer::type_id::create("s2p_sqr_0", this);
end
```

Instead of `new()`
Instance Paths - name

You’re going to call me WHAT!?
axi_fabric_env – How to Name It?

virtual class axi_fabric_base_test extends uvm_test;

    axi_fabric_env axi_fabric_env_0;

    `uvm_component_utils(axi_fabric_base_test)

    function new(string name, uvm_component parent);
        super.new(name,parent);
    endfunction : new

    extern virtual function void build_phase(uvm_phase);

endclass : axi_fabric_base_test

function void axi_fabric_base_test::build_phase(uvm_phase);
    super.build_phase(phase);
    axi_fabric_env_0 = axi_fabric_env::type_id::create(.name("CRAZY_ENV"),
                .parent(this));

endfunction : build_phase
axi_fabric – Factory Overrides

```verbatim
function void axi_fabric_factory_t::test::build_phase(uvm_phase phase);
    uvm_factory f;
    super.build_phase (phase);
    set_inst_override_by_type(.
        relative_inst_path("axi_fabric_env_0.*agent*0.*.sequencer.*"),
        original_type(axi_master_xactn::get_type()),
        override_type(axi_hword_xactn::get_type()));
    set_inst_override_by_type(.
        relative_inst_path("*.agent*1.axi_master_sequencer.*"),
        original_type(axi_master_xactn::get_type()),
        override_type(axi_word_xactn::get_type()));
    set_inst_override_by_type(.
        relative_inst_path("CRAZY_ENV.*agent*2.axi_master_sequencer.*"),
        original_type(axi_master_xactn::get_type()),
        override_type(axi_byte_xactn::get_type()));
    uvm_factory::get();
    f.print(.all_types());
endfunction : build_phase
```
Objections in UVM

• No news is good news
• By default no one objects!
  – Hence test ends at ZERO time if not taken care of!
Scalable End Of Test via “Objection”

http://www.cvcblr.com/blog/?p=414
virtual task rand_test::main_phase(uvm_phase phase);
    super.main_phase(phase);
    phase.raise_objection(this);
    s2p_seq1_0 = s2p_seq::type_id::create("s2p_seq1");
    s2p_seq1_0.start(env_0.agent_0.s2p_sequencer_0);
    phase.drop_objection(this);
endtask : main_phase

http://www.cvcblr.com/blog/?p=414
Debugging Phase Execution

- UVM has built-in debug feature for Phases
- vw_uvm_sim +UVM_PHASE_TRACE
- Look for ID - [PH/TRC/]*

UVM_INFO @ 0: reporter
[PH/TRC/SKIP] Phase
'uvm.uvm_sched.main' (id=284)
No objections raised, skipping phase
Advanced Debug Methodology – Hangs?

```verilog
phase.drop_objection(this);
`uvm_info (get_name, "End of test", UVM_MEDIUM);
endtask: main_phase

task apb_test_1::dbg_eot(uvm_phase phase);
forever begin : fe
  #100;
  phase.phase_done.display_objections();
end : fe
endtask : dbg_eot
```

Built-in `display_objections`

http://www.cvcblr.com/blog/?p=681
Advanced Debug Methodology – Hangs?

The above indicates there are 3 folks opposing it with clear pointers to who they are.

http://www.cvcblr.com/blog/?p=681
Phase Jumping

• UVM allows phase jumps
  – Forward
  – Backward
• Handy if a test wants to “run” few times
• Multiple runs through:
  – reset-cfg-main-shutdown
  – cfg-main-shutdown
  – reset-main
Subsystem Test

- phase_ready_to_end callback

```cpp
extern virtual function void build_phase(uvm_phase phase);
extern virtual task reset_phase(uvm_phase phase);
extern virtual task configure_phase(uvm_phase phase);
extern virtual task main_phase(uvm_phase phase);
extern virtual task post_shutdown_phase(uvm_phase phase);
extern virtual function void phase_ready_to_end (uvm_phase phase);

endclass: vl_subsys_test
```
Subsysytem Test

- Mimics reset, config via delays
- Uses a virtual sequence in main_phase
Innings Break!

function void vl_subsys_test::phase_ready_to_end(uvm_phase phase);
    super.phase_ready_to_end(phase);
    if (phase.get_imp () == uvm_post_shutdown_phase::get()) begin : end_of_one_phase_run
        `uvm_info (vl_id, "Reached post_shutdown_phase, checking for num_jumps", UVM_MEDIUM)
        if (this.num_jumps > 0) begin : more_runs_needed
            `uvm_info (vl_id,
                $sformatf("Jumping back to pre_reset_phase as num_jumps is: %0d", this.num_jumps), UVM_MEDIUM)
            phase.jump(.phase(uvm_pre_reset_phase::get()));
            this.num_jumps--;
        end : more_runs_needed
        else begin : end_of_marathon
            `uvm_info (vl_id, "End of a marathon run with few jumps!", UVM_MEDIUM)
        end : end_of_marathon
    end : end_of_one_phase_run
endfunction : phase_ready_to_end
Second Innings

Match not over yet!

```verbatim
function void vl_subsys_test::phase_ready_to_end(uvm_phase phase);
  super.phase_ready_to_end(phase);
  if (phase.get_imp () == uvm_post_shutdown_phase::get()) begin : end_of_one_phase_run
    `uvm_info (vl_id, "Reaching post_shutdown_phase, checking for num_jumps", UVM_MEDIUM)
    if (this.num_jumps > 0) begin : more_runs_needed
      `uvm_info (vl_id,
        `$sformatf("Jumping back to pre_reset_phase as num_jumps is: %0d", this.num_jumps), UVM_MEDIUM)
      phase.jump (.phase(uvm_pre_reset_phase::get()));
      this.num_jumps--;
    end : more_runs_needed
    else begin : end_of_marathon
      `uvm_info (vl_id, "End of a marathon run with few jumps!", UVM_MEDIUM)
      end : end_of_marathon
    end : end_of_one_phase_run
  endfunction : phase_ready_to_end
```
Phase Synchronization

• By default, all components must allow all other components to complete a phase before all components move to next phase

VIP 1: reset configure main shutdown

VIP 2: reset configure main shutdown

VIP 3: reset configure main shutdown

---

2/29/2016

UVM Domain

- Domains are collections of components that must advance phases in unison
  - By default, no inter-domain synchronization

VIP 1: reset configure main shutdown
VIP 2: reset configure main shutdown
VIP 3: reset configure main shutdown

Domain A

Domain B

time
Phases - Guidelines

- Use correct phasing
- Use objections
- Learn UVM CLP for debug
- Advanced debug – display_objections
- Jumps are cool!
  - And useful 😊
- Can also use domains!
  - Advanced topic, call us again 😊
UVM Config DB
Configuration Database
Field Name vs. Value

• Typically field_name and value are kept the same
• Not a must, set and get must agree on the field_name
Field Name vs. Value

Field name

value

function void s2p_agent::build_phase(uvm_phase phase, 
super.build_phase(phase);
if (!uvm_config_db#(uvm_active_passive_enum).get(this, 
"HAS_active", 
is_active)) begin : def_val_for_is_active
`uvm_warning(get_name,$sformatf("No override for is_active: Using default is_active as:%s", 
this.is_active.name));
end : def_val_for_is_active

`uvm_info(get_name(),$sformatf("is_active is set to %s",this.is_active.name),UVM_MEDIUM);
Field Name vs. Value

function void s2p_env::build_phase(uvm_phase phse);
super.build_phase(phase);
agent0 = s2p_agent::type_id::create(.name("agent0"),
    .parent(this));

uvm_config_db#(uvm_active_passive_enum)::set(.ctxt(this),
    .inst_name("*"),
    .field_name("WAS_active"),
    .value(UVM_ACTIVE));

endfunction : build_phase
What if “set”, no “get”?

- Usually a BIG problem!
- Hard to detect bugs (TB bugs)
- uvm_component has handy debug functions for this!
check_config_settings

extern virtual task main_phase(uvm_phase phase);

function void start_of_simulation_phase(uvm_phase phase);
this.check_config_usage();
endfunction : start_of_simulation_phase

dendcode : rand_test

# ________________________________________________________________
# # 0.00 ns UVM_INFO | End Of Elaboration | reporter | Questa UVM | verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv : 272
# 0.00 ns UVM_INFO | :::: The following resources have at least one write and no reads :::: | uvm_test_top | CFGNRD |
# : 0
# num_masters [^uvm_test_top\..env_0\..*$/] : (int) 3
# -
#  -------------------
#  uvm_test_top.env_0 reads: 0 @ 0.00 ns writes: 1 @ 0.00 ns
# # 0.00 ns UVM_INFO | Run Phase is Running ..
check_config_settings

num_masters

Set (write)

NO get (read)
task rand_test::main_phase(uvm_phase phase);
    phase.raise_objectection(this);
    `uvm_info("Rand Test","Test is running...",UVM_LOW)

uvm_config_db#(int)::set(.cntxt(this),
    .inst_name("*"),
    .field_name("num_jumps"),
    .value(5));

this.print_config(.recurse(1), .audit(1));

s2p_seq01 = s2p_rand_seq::type_id::create(.name("s2p_seq"),
    .parent(this));
this.s2p_seq01.start(env_0.agent0.sequencer);
Command Line Processor
| +UVM_MAX_QUIT_COUNT=2                                                                 | Quit after N number of errors  |
|                                                                                  | Default ➔ No limit            |
|                                                                                  | Only aware of `uvm_error       |
|                                                                                  | SVA action blocks, DPI calls should use uvm_error |
UVM CLP Options for Debug

- Built into UVM library at the major points of execution
- Dump important runtime data into log for debug or post-processing
- Can be activated from command line options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+UVM_PHASE_TRACE</td>
<td>Turns on tracing of phase execution</td>
</tr>
<tr>
<td>+UVM_OBJECTION_TRACE</td>
<td>Turns on tracing of objection activities</td>
</tr>
<tr>
<td>+UVMRESOURCE_DB_TRACE</td>
<td>Turns on tracing of resource DB access (get &amp; set)</td>
</tr>
<tr>
<td>+UVM_CONFIG_DB_TRACE</td>
<td>Turns on tracing of configuration DB access</td>
</tr>
</tbody>
</table>
+UVM_CONFIG_DB_TRACE

# Set configuration database traces

```verilog
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg_sv(215) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.2
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg_sv(217) @ 0: reporter [Questa UVM] questa_uvm::init(+struct)
# UVM_INFO @ 0: reporter [CFGDB/SET] Configuration 'uvm_test_top.env_0.agent0.s2p_if_0' (type virtual s2p_if) Set by = (virtual s2p_if) /top/s2p_if_0
# UVM_INFO @ 0: reporter [CFGDB/SET] Configuration 'uvm_test_top.env_0.*.is_active' (type enum bit uvm_pkg.uvm_resource_db.uvm_resource_db_12.m_show_msg.uvm_active_passive_enum) set by uvm_test_top.env_0 = (enum bit uvm_pkg.uvm_resource.uvm_resource_12.convert2string.uvm_active_passive_enum) UVM_ACTIVE
# UVM_INFO @ 0: reporter [CFGDB/GET] Configuration 'uvm_test_top.env_0.agent0.s2p_if_0' (type virtual s2p_if) read by uvm_test_top.env_0.agent0 = (virtual s2p_if) /top/s2p_if_0
# UVM_INFO @ 0: reporter [CFGDB/GET] Configuration 'uvm_test_top.env_0.agent0.is_active' (type enum bit uvm_pkg.uvm_resource_db.uvm_resource_db_12.m_show_msg.uvm_active_passive_enum) read by uvm_test_top.env_0 = (enum bit uvm_pkg.uvm_resource.uvm_resource_12.convert2string.uvm_active_passive_enum) UVM_ACTIVE
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_resource_db_sv(121) @ 0: reporter [CFGDB/GET] Configuration 'uvm_test_top.recording_detail' (type reg signed[4095:0]) read by uvm_test_top = null (failed lookup)
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_resource_db_sv(121) @ 0: reporter [CFGDB/GET] Configuration 'uvm_test_top.recording_detail' (type int) read by uvm_test_top = null (failed lookup)
# UVM_INFO @ 0: reporter [RNTST] Running test rand_test...
```

• ID: \texttt{CFGDB/GET, CFGDB/SET}

```
# UVM_INFO ..//tb_src/s2p_fcov.sv(48) @ 5430: uvm_test
#
# ---- UVM Report Summary ----
#
# ** Report counts by severity
# UVM_INFO : 203
# UVM_WARNING : 0
# UVM_ERROR : 0
# UVM_FATAL : 0
# ** Report counts by id
# [CFGDB/GET] 141
# [CFGDB/SET] 2
# [Questa UVM] 3
# [RNTST] 1
# [Rand_TEST] 1
# [Rand_Test] 1
# [SBRD] 21
# [UVMTOP] 1
# [agent0] 1
# [driver] 9
# [get interface] 1
```
• Displays “added objections” for raise_object

+UVM_OBJECTION_TRACE

- Displays “added objections” for raise_object

+UVM_OBJECTION_TRACE
• Displays “dropped objections” for drop_object
• ID: OBJTN_TRC
Popular, Simple Options

- `+UVM_TESTNAME=my_test`
  - Executes "class my_test extends from uvm_test"
  - Should be registered with factory
    - `\uvm_component_utils (my_test)`
    - Overrides value (if present) in code:
      - `run_test ("default_test")`
  - Only one test is run at a time
UVM_TESTNAME - Multiple

- Multiple values – ignored, warning issued
- `vw_uvm_sim +UVM_TESTNAME=first_test`  
  `+UVM_TESTNAME=second_test`

```
Multiple (2) +UVM_TESTNAME arguments provided on the command line.  
'first_test' will be used.

Provided list: first_test, second_test
```
Verbosity Setting

- `vw_uvm_sim +uvm_set_verbosity`

**Template**

```
vw_uvm_sim
+uvm_set_verbosity=<comp>,<id>,<verbosity>,<phase|time>,<offset>
```

**Example**

```
vw_uvm_sim
+uvm_set_verbosity=uvm_test_top.env0.agent1.*,_ALL_,UVM_FULL,time,800
```
Severity Override in CMD Line

- `vw_uvm_sim +uvm_set_severity`

**Template**
```
vw_uvm_sim
+uvm_set_severity=<comp>,<id>,<orig_severity>,<new_severity>
```

**Example**
```
vw_uvm_sim
+uvm_set_severity=uvm_test_top.env_0.*,SBRD,UVM_INFO,UVM_WARNING
```
Changing “action” on Messaging

- `vw_uvm_sim +uvm_set_action`

**Template**

```
vw_uvm_sim
+uvm_set_action=<comp>,<id>,<severity>,<action[|action]>
```

**Example**

```
vw_uvm_sim
+uvm_set_action=uvm_test_top.env_0.*,_ALL_,UVM_INFO,UVM_NO_ACTION
```
UVM CLP - Guidelines

• Change verbosity, severity, error count on CMD line
• Avoid factory overrides via CMD line
  – Hard to regress
  – Verification becomes spread across UVM, Perl/Scripts
Summary

• Runtime tricks & tips in UVM
• Several built-in hooks in UVM for run-time debug
• Look for our **DVRules** product to flag these @ [http://www.verifworks.com](http://www.verifworks.com)
• Use our DVCreate tools to generate quality UVM
Thank You!